Invisible and Forgotten: Zero-Day Blooms in the IoT

Kartik Palani*, Emily Holt†, and Sean Smith†

*Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

†Department of Computer Science
Dartmouth College

***Abstract*—In the IoT, massive distribution and long physical lifetimes will disrupt the "penetrate and patch" security paradigm that helps mitigate the consequences of the vulnerabilities endemic in individual systems. In this paper, we examine what will happen in the IoT if we build its systems the same way. We collect data and model the vulnerability blooms and patching delays in historical systems. We present the models and discuss future IoT networks where similar blooms happen but patching does not. We discuss initial results, and our plans to extend the models to look more deeply at these questions in our future work.**

I. Introduction

The current "Internet of computers" (*IoC*) is composed of devices rife with security vulnerabilities. To help compensate for these endemic vulnerabilities, the current IoC depends on a "penetrate and patch" security paradigm. As holes are discovered, software is patched and eventually retired; as new attacks emerge, new signatures are pushed to anti-virus software and firewall rules are updated.

We already see IoT devices being built with the same endemic holes: for example, police body cameras too weak to run anti-virus themselves ship with the Conficker virus already installed [1]; analysts report that "over 750,000 phishing and spam messages" have already been sent "from more than 100,000 household devices—televisions, wi-fi routers, and fridges" whose internal computers have been taken over as bots [2]. We also already see the nature of the devices and their distribution in the IoT disrupting the penetrate and patch paradigm—e.g., [3]. In this paper, we seek to examine the security impact of this disruption, and also to provide a foundation to examine the effectiveness to potential solution strategies.

Section II considers some principal ways in which the nature of the IoT may disrupt the old security paradigm. Section III then presents our approach to the problem. Section IV looks at history to develop some mathematical models of zero-day blooms. Section V examines some ways to quantitatively describe the security health of a large system such as the IoC or IoT. Section VI presents the model we built to examine the effect the changes from IoC to IoT have on security health from zero-day blooms. Section VII presents our current results. Section VIII discusses future work: how we plan to use this and other models to tackle the problem.

II. IoC to IoT

Many aspects of the transition from the IoC to the IoT have the potential to disrupt the IoC's security paradigms. In this section, we discuss some principal ones of concern.

a) Invisibility: As responsible users know, it can be very hard to stay on top of keeping software updated; as system administrators know, it can very hard to convince users to be responsible. When computers stop looking like computers, we hypothesize the problem will become much worse.

b) Lifetimes: Physical devices such as appliances and light-switches will likely live much longer than laptops and personal computers; e.g., washing machines may last more than a decade, and power grid equipment may last many decades. Thus, we hypothesize that unpatched software will persist much longer. Exacerbating the problem is the fact that devices may outlive their original ownership period—and may even outlive the company responsible for maintaining the software.

As an example already of the hazards of device lifetime and invisibility, radiology machines with embedded computer containing old and unpatched Windows operating systems led to an infection paralyzing a hospital's IT system [4]; as another example, air traffic systems based on even older Windows 3.1 led to the November 2016 shutdown of the Orly airport [5].

c) Patchability: We hypothesize that devices in the IoT will be harder to patch than devices in the IoC. One contributing factor is reduced network connectivity—remote devices may be hard to reach over the network; to patch the well-publicized security flaw in Jeep Cherokees, Chrysler needed to physically mail USB drives to owners. Another factor may be the fact the device software may be in unchangeable ROM or FLASH, leading to some analysts warning about the emergence of *forever-days*. A 2014 Kaspersky Labs' study [6] showed that the CVE-2010-2568 vulnerability that was exploited by Stuxnet still remained un-patched on over 19 million computers worldwide.

d) Device Weakness: We hypothesize that the security contributions of anti-virus will decrease in the IoT, as devices will be too weak to run AV (compounded by the factors above reducing the degree and effectiveness of patching).

e) Consequences of Compromise: We hypothesize that the intimate connection of IoT devices to physical infras-

tructure will increase the damage from successful compromise. Exploding gasoline tanks [7], radio broadcasts crippling automobiles in transit [8], and blacking out northeast North America [9] are all much worse than not being able to buy things on Amazon for three days.

f) Population Scale: We hypothesize that the increased size of the IoT (e.g., many tens of billions, instead of millions) will also reduce the effectiveness of penetrate and patch. The number of vulnerable systems will be larger, and the speed with which patches can propagate may in fact be longer.

III. Approaching the Problem

An absolute quantification of security, although useful in providing guarantees to user, can not be achieved. One can not claim that the IoT architecture is x% secure. However, a relative quantification is possible. A relative quantification would suggest a certain improvement or a certain decline in the security of a system. While current strategies to quantify the security of a system come from pen-testing teams and security experts trying to break into a system, a new trend of modeling security has emerged in the past decade—evaluation approaches based on security metrics based and modeling the attacker (script kiddie, serious hackers and nation states) along with the cause-effect relationships between actions. In the future we hope to build models that take into account the ability of the attackers. In our ongoing work, we aim to relatively quantify the changes in security patterns and metrics from the IoC to IoT.

IV. Modeling Zero-Day Blooms

We want to be able to predict the growth rate of zero days in unpatched software. Building on the analysis of Bilge and Dumitras [10], we want to determine the window between first release of an exploit and release of its patch. For generating our model, we consulted methods from Nicol et al [11].

To build these models, we look at the past: the *Common Vulnerabilities and Exposures (CVE)* list maintained by MITRE. We generated a list of all reports containing specific user-provided keywords. We then parsed the list of reports to obtain the aggregate number of vulnerabilities per year relating to this keyword. We used the number of vulnerabilities reported per year to create a curve of best fit (linear or polynomial). We then used this curve of best fit to generate projections for the years 2020 and 2025.

Figure 1 shows the curve for server-side software (keyword 'server'); Figure 2 shows the curve for browser-side software (keyword 'browser'); and Figure 3 shows the curve for device software (keyword 'device'). We looked at these because, in consultation with vulnerability analysts, we felt that browsers were the new servers, and devices the new browsers. We decided to use Figure 3 because this curve projects the future number of vulnerabilities reported that will relate to device software. We predict vulnerability reports for device software—specifically for software in IoT devices—will flare-up over the next 10-15 years.

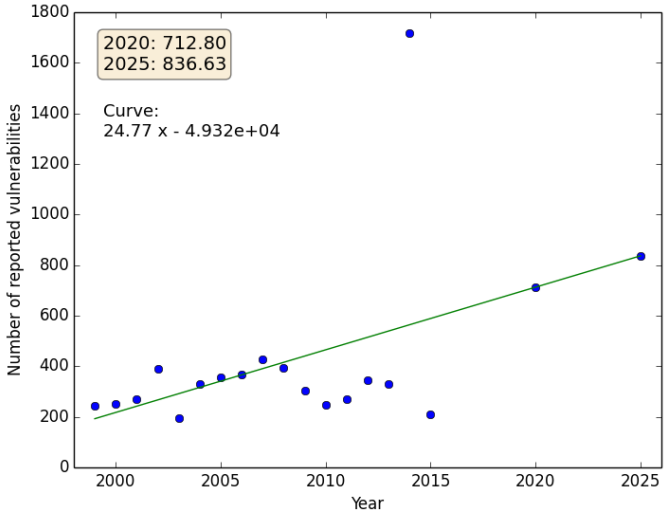

Fig. 1. Server vulnerabilities reported in CVE through 2015—and projected to 2025

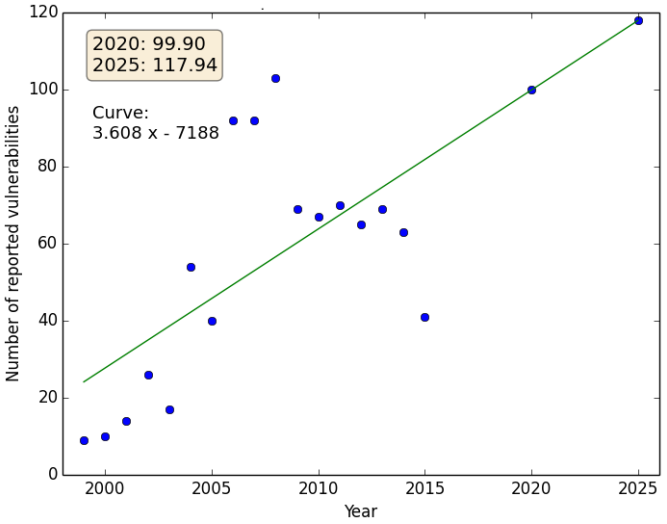

Fig. 2. Browser vulnerabilities reported in CVE through 2015—and projected to 2025

One challenge to this approach (of merely measuring reported CVE vulnerabilities) is that vulnerability reports and subsequent blooms often may result from concerted efforts from dedicated researchers and inspired followers driving the "Month of Bugs" phenomenon. For example, HD Moore launched the Month of Browser Bugs (July 20006) in an effort to raise awareness about common cybersecurity threats [12]. A group of dedicated security researchers launched the "Month of Kernel Bugs" in November the same year [13]. Researchers were spurred by advances in the security research community, specifically the free availability of "fuzzing" tools, as well as the incorporation of kernel exploits into the free Metasploit penetration testing tool. Similarly inspired, Stefan Esser commenced the Month of PHP Bugs in March of 2007 [14].

Shahzad et al [15] found that since 2004, there has been

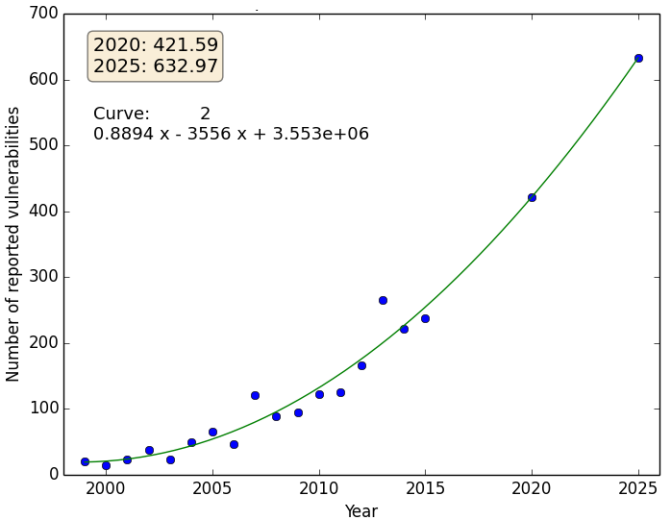

Fig. 3. Device vulnerabilities reported in CVE through 2015—and projected to 2025

a decrease in the proportion of the vulnerabilities exploited before their public disclosure date, and an increasing trend in the proportion of vulnerabilities exploited on the date of their public disclosure—illustrating that hackers are becoming increasingly active. Shahzad et al also observed that "the percentage of remotely exploitable vulnerabilities has gradually increased to over 80% of all the vulnerabilities," which is important for our research on IoT devices. If attacks are trending towards being remotely executed, and devices are becoming increasingly prevalent in the world if IoT, this has big implications for security risk of customers. IoT devices are also becoming increasingly invisible. Thus customers are at an increasingly greater risk with decreasing awareness of their risk.

V. Measuring Security Health

We want to reason about the security impact of some given number of unpatched zero-days on the IoT.

We start by considering an individual device. At any given time, it may be in one of three states:

- It may have no known vulnerabilities.
- It may have known vulnerabilities.
- It may be compromised.

An attack on a vulnerable device will lead to compromise, and a compromised device may then launch attacks on the devices to which it is connected. A patch issued to a vulnerable or compromised device will return it to healthy; a healthy device will become vulnerable according to a stochastic process derived from Figure 3. (In this initial model, for simplicity, we treat "vulnerabilities" of a device as an aggregate set instead of separate items with separate patches.)

To consider the health of the overall IoT, we might initially measure (in our model) the percentage of devices that are compromised. However, this measure alone will not capture the notion of resilience to attack: just because the devices are not currently compromised does not mean that a small focused attack cannot cause disaster. So, instead, we will measure over time:

- the percentage of devices in the system that will become compromised
- versus the percentage of devices against which an attack is launched.

A low, flat curve indicates resilience; one that grows quickly indicates trouble.

(In future work, we will also consider the physical impact of a compromise.)

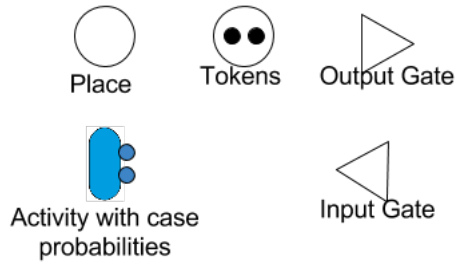

Fig. 4. Graphical Representation of Stochastic Activity Networks

VI. Modeling Health in the IoT

We build our model on the Stochastic Activity Networks (SANs) [16] formalism. SANs are an extension to Petri Nets. SANs use graphical primitives to provide high-level formalism which allows for detailed specification of performance models. It contains four most important components:

- *Places* represent the state of the modeled system. Places contain *tokens* that represent the *marking* of the place. Places are like variables that contain the state or "value" of the system. Markings may either represent the number of objects, as in the number of cars that need to be washed; or markings may represent an object of a certain type, as in the priority of a task. This allows for a lot of flexibility.
- *Activities*, timed or instantaneous, are actions that take place in the modeled system over a period of time. Each timed activity has a time distribution function associated with it. When an activity fires, it causes a *transition* in the state of the system and thus changes the marking of a place.
- *Input gates* control the firing of activities and define the changes in marking that will occur after a transition takes place. Each input gate has an enabling predicate and a function. While the enabling predicate decides whether an activity must fire or not, the function defines the marking changes that must occur post activity completion.
- *Output gates* are also used to define complex completion functions. They only differ from input gates in that they can only be associated with one case if the activity has multiple cases it can choose from after it completes.

To model the IoT itself, we'll consider the following SAN:

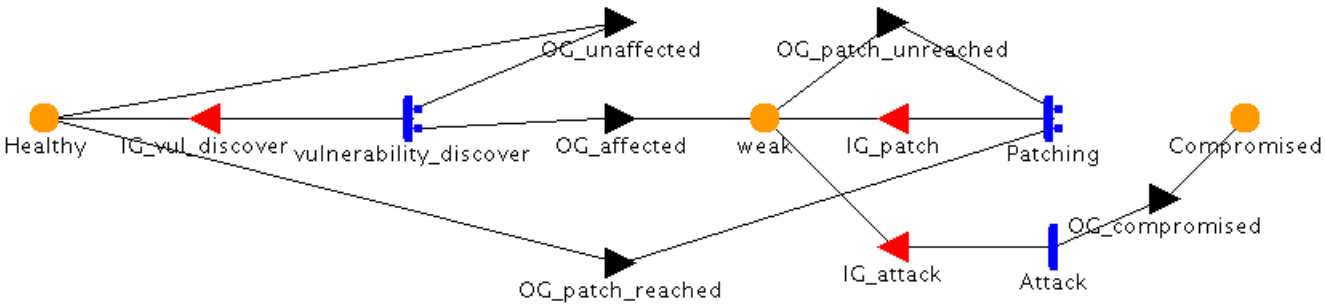

Fig. 5. Stochastic Activity Networks for system health

- There are three places representing the three states of the model: healthy, weak or compromised.
- The markings in each place represent the number of devices in that state.
- The gates govern the transitions and contain the functions that get executed when an activity fires.
- The $vulnerability\_discover$ activity is the rate at which vulnerabilities get discovered by researchers and the security community. When a vulnerability gets discovered, there is a chance that the system is affected by that vulnerability—in which case, it changes state from healthy to weak. There is also a chance that the vulnerability does not affect the system—in which case, it remains healthy.
- The $patching$ activity is the rate at which patches are sent out after a vulnerability has been discovered. A patch might either reach the system or not. If the system is patched it goes from being weak to healthy; if the patch doesn't reach the system it remains weak. Here we define a *patchability* constant p as the probability that a patch successfully reaches a peripheral device.
- The $attack$ activity defines an attack on a system before it is patched. We consider that an attack on a known vulnerability will always succeed—thus going from the weak state to the compromised state.

Figure 5 shows the graphical representation of the SAN model that we built. We hope to use this model as a starting point to answer questions regarding the transition form IoC to IoT.

High level formalisms like SANs allow for ease in specification of large-scale systems. With a large model comes a large state space which may take unreasonably long to solve using analytical solutions. Discrete event simulation can be used to solve for such cases where the state spaces are too large. One drawback of discrete event simulation is the fact that if the desired measure of solving a large model is based on a 'rare' event then the result may not be accurate.

In our case, we use discrete event simulation to solve the model described in this section. The fast growth of the Internet of Things population, the large number if vulnerabilities discovered every year and the even larger number of attacks on such connected environments makes it such that there are no rare events in our model. Thus, simulation is a dependable choice.

VII. Initial Results

In this section we present the initial results of our analysis. We study the system with a growth in the number of IoT connected devices, an increase in vulnerabilities discovered and a raise in the number of attacks on the devices.

A. Parameter Choices

In our analysis we consider two variable inputs: the rate of growth of the number of devices in the Internet of Things and the rate at which vulnerabilities are discovered. The vulnerability growth profile is based on the CVE study described in Figure 3, which shows a steady rise in the number of vulnerabilities discovered every year. The profile for the increase in population size of the IoT is based on the intelligence study in Greenough [17]. This profile is shown in Figure 6. The selection of the input parameters is based on the intuition that as the number of devices in a highly interconnected environment grows, discovered vulnerabilities have a larger surface on which to spread.

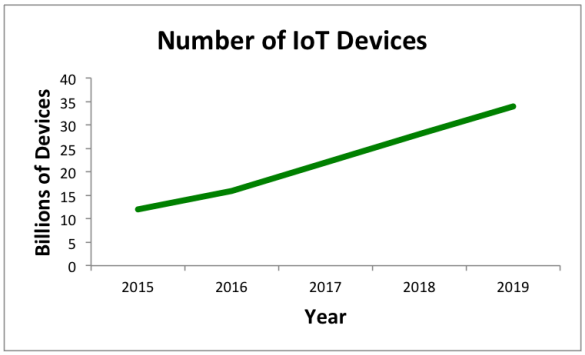

Fig. 6. Approximation of IoT population growth based on studies in Greenough [17]

As mentioned earlier, we are interested in the relative quantification of security; hence our other parameters are conservative assumptions that help us understand the spread of attacks in a highly connected environment. We assume that

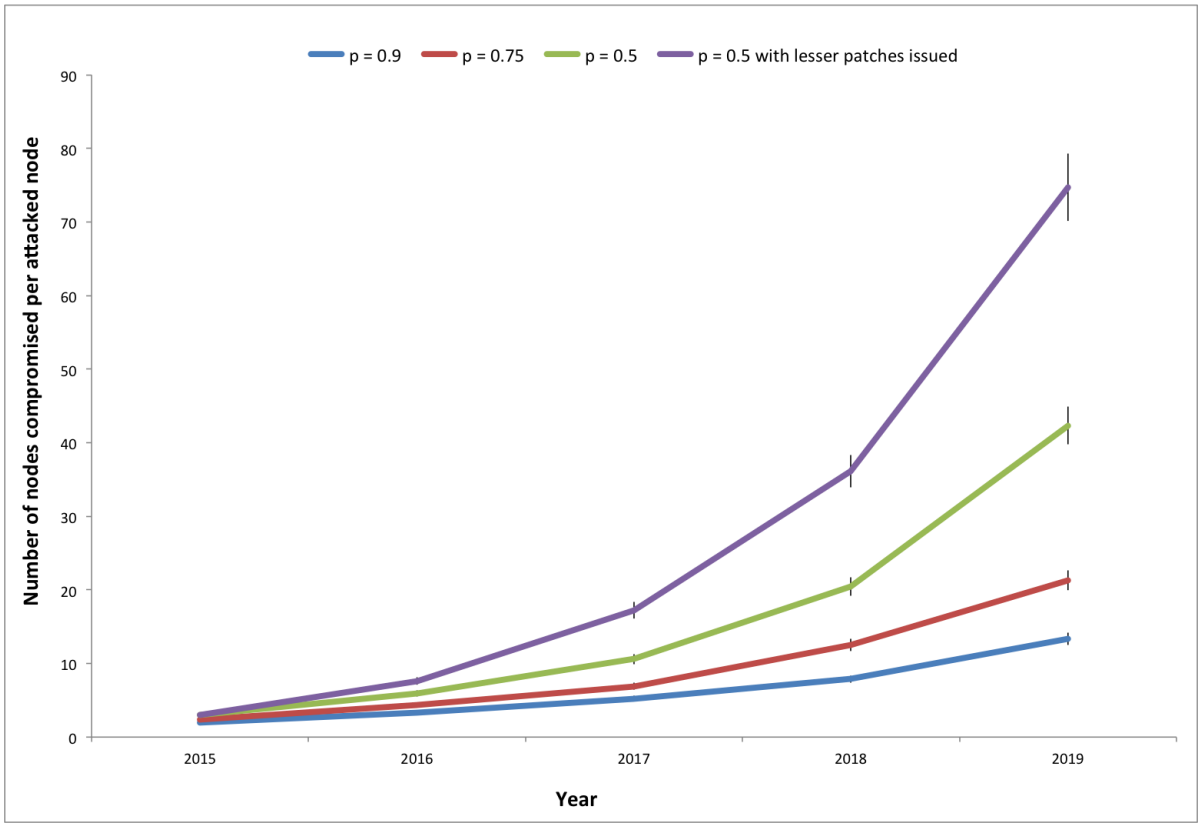

Fig. 7. The number of nodes that get compromised per attacked node over time for different patchability constants (p), which is the probability that a patch successfully reaches a node.

every year an attack is launched against 10% of the devices in the population. This is the $attack\_rate$ used in the model. We also provide a study for two cases of vulnerability patches: the first study assumes that there is a patch for every vulnerability that is discovered that year, the second study assumes that only 90% of the vulnerabilities discovered have patches. These provide the $patch\_rate$ value to the model. We provide the results for both these cases in this section.

B. Results

Figure 7 describes the results we obtained. The plot aims to find out how many nodes are compromised when one node is attacked in a highly connected environment.

The first study we performed was to see the impact of change in the patchability constant. For this, we assume an ideal scenario where for every vulnerability that is discovered, a patch is issued. As we can see, even when 90% of the patches reach the nodes, there are still multiple nodes that get affected when a single node is attacked. As can be seen for the cases where $p = 0.75$ and $p = 0.5$, as the patchability becomes worse, more nodes get affected by a single failed node. This shows the high amount of dependence in the IoT mesh network scenario.

The second study not only retards the patchability but also considers that only 90% of the vulnerabilities have patches issued/discovered. In this case, for every device that is attacked, potentially, more than 80 devices will be compromised.

The main goal of this study is to provide knowledge of the fact that with the Internet of Things, the probability that a patch successfully reaches every device is going to reduce drastically. As the patchability gets worse, the highly interconnected and dependent nature of the IoT would mean that a single exploitation of a vulnerability on a single node would lead to various other nodes becoming potentially compromised. While the results of this study seem intuitive, the study motivates the problem and provides a starting point for research in this direction, which is our goal. We also hope that when we use our model to study other IoT communication architectures with different levels of interdependence, we will have a better understanding of the schemes that need to be implemented in order to avoid such spreads of the attack.

C. Computation Times

To get the above results, we used a discrete event simulation software called Möbius. The simulator uses a Lagged Fibonacci generator as the pseudo random number generator to traverse the model. The analysis has been performed on a machine equipped with a 2.9 GHz Intel Core i7 processor and 8GB of RAM. The simulator only utilizes one core of the processor. The computation for each patchability setting takes about 10-15 minutes.

VIII. Conclusions and Future Work

Vulnerabilities—from zero-days to forever-days—are rampant in today's embedded systems. Patching these vulnerabilities is already incredibly complicated, and as we transition from the IoC to the IoT, patching these systems will be progressively more difficult—not only because of the growing number of devices affected by the discovered vulnerabilities, but also because devices may outlive the vendors responsible for their technology or their maintenance.

In this paper we presented this problem, and a preliminary model to help examine this problem. In our initial results here, we examine net security health of a large system when the scale increases but patchability lags.

We plan to continue this work first by extending the model to look at more sophisticated topologies and more sophisticated threat models. For the former, the emerging IoT offers many competing architectural visions, from home gateways, to multiple home gateways, to Cloud avatars, to meshes. We want to revise the model to reflect the security implications of these various topological choices—and of having a superposition of several of them. For the latter, we plan to revise the model to consider heterogeneous vulnerability/patching patterns and lags, topological implications on patching and attacks, emergence of thingbots, and net physical impact of attacks and thingbots.

However, our long-term goal is not to forecast doom but rather to avoid it. To that end, a model to analyze vulnerability impact in the IoT will also extend to evaluating the efficacy and performance impact of various proposed mitigating techniques. For example: what if instead of pushing patches to individual devices, we inserted verifiable protocol filters at key points in the network? What would be the impact of building in "telomeres"—features that cause devices to automatically die after a certain point, or variations such as after-last-update? What about "dumb grids": selectively disabling various features of smart devices? Can N firewalls do almost as well as $100N$ formally-verified devices? This is where we plan to take the model next.

Acknowledgements

This work was sponsored in part by the US Department of Energy under grant agreement DE-OE0000780. The views expressed are those of the authors.

References

[1] C. Cimpanu, "Police Body Cameras Shipped with Pre-Installed Conficker Virus," *Softpedia*, November 2015.

[2] P. Belton, "Is Your Toaster a Silent Recruit in a 'Thingbot' Army?" *BBC*, Feburary 2015.

[3] V. Zhang, "High-Profile Mobile Apps At Risk Due to Three-Year-Old Vulnerability," *TrendLabs Security Intelligence Blog*, December 2015.

[4] Anonymous Clinician, "Personal communication."

[5] P. Longeray, "Windows 3.1 Is Still Alive, And It Just Killed a French Airport," *Vice News*, November 2015.

[6] Kaspersky Lab, "The Echo of Stuxnet," *SecureList.com*.

[7] K. Wilhoit and S. Hilt, *The GasPot Experiment: Unexamined Perils in Using Gas-Tank-Monitoring Systems*. TrendLabs, 2015.

[8] C. Vallance, "Car hack uses digital-radio broadcasts to seize control," *BBC*, July 2015.

[9] K. Poulsen, "Tracking the Blackout Bug," *Security Focus*, April 2004.

[10] L. Bilge and T. Dumitras, "Before We Knew It: An Empirical Study of Zero-Day Attacks In The Real World," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, 2012.

[11] D. M. Nicol, W. H. Sanders, and K. S. Trivedi, "Model-Based Evaluation: From Dependability to Security," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, 2004.

[12] S. M. Kerner, "The Month of The Browser Bugs Begins," *InternetNews.com*, July 2006.

[13] R. Mogull, "Learn from 'Month of Kernel Bugs'," *Gartner*, November 2006.

[14] B. Prince, "Month of PHP Bugs Begins," *eWeek*, March 2007.

[15] M. Shahzad, M. Z. Shafiq, and A. X. Liu, "A Large Scale Exploratory Analysis of Software Vulnerability Life Cycles," in *34th International Conference on Software Engineering*.

[16] W. H. Sanders and J. F. Meyer, "Stochastic Activity Networks: Formal Definitions and Concepts," in *Lectures on Formal Methods and Performance Analysis*. Springer, 2001, pp. 315–343.

[17] J. Greenough, "The Internet of Everything: 2015," *Business Insider Intelligence*, April 2015.